```
clear all; close all; clc;
tic;  % Inicia cronometraje

%---------------------------------------------------------------------------
% Declaración de variables simbólicas
%---------------------------------------------------------------------------
% Para robot cartesiano, las variables representan desplazamientos lineales
syms th1(t) th2(t) th3(t) t            % Desplazamientos en cada eje
syms th1p(t) th2p(t) th3p(t)           % Velocidades de cada desplazamiento
syms th1pp(t) th2pp(t) th3pp(t)         % Aceleraciones de cada
desplazamiento
syms m1 m2 m3                          % Masas de cada eslabón
syms Ixx1 Iyy1 Izz1 Ixx2 Iyy2 Izz2 Ixx3 Iyy3 Izz3  % Momentos de inercia de
cada eslabón
syms l1 l2 l3 lc1 lc2 lc3              % l: recorrido máximo; lc: distancia
al centro de masa
syms pi g a cero                       % Constantes físicas y auxiliares

%---------------------------------------------------------------------------
% Vectores articulares: coordenadas, velocidades y aceleraciones
%---------------------------------------------------------------------------
Q   = [th1; th2; th3];       % Coordenadas articulares (desplazamientos)
Qp  = [th1p; th2p; th3p];     % Velocidades articulares
Qpp = [th1pp; th2pp; th3pp];   % Aceleraciones articulares

%---------------------------------------------------------------------------
% Configuración del robot: Todas las juntas son prismáticas
% 1 => junta prismática, 0 => junta rotacional
%---------------------------------------------------------------------------
RP = [1 1 1];               % Robot cartesiano de 3 GDL
GDL = size(RP,2);           % Número de grados de libertad

%---------------------------------------------------------------------------
% Definición de la cinemática directa
% Adaptación para un robot cartesiano:
% - Las variables (th1, th2, th3) son desplazamientos lineales
% - Se definen las posiciones y orientaciones de cada "articulación"
%---------------------------------------------------------------------------

% Articulación 1: Desplazamiento en Z (por ejemplo)
P(:,:,1) = [0; 0; l1];
% La matriz de rotación se define para orientar el marco; en este caso se
usa -90° en x
R(:,:,1) = [ 0,  0,  1;
             0,  1,  0;
            -1,  0,  0];

% Articulación 2: Desplazamiento en Z
P(:,:,2) = [0; 0; l2];
```

```
% Se define una rotación de -90° en y para reorientar el sistema de
referencia
R(:,:,2) = [ 1,  0,  0;
             0,  0, -1;
             0,  1,  0];

% Articulación 3: Desplazamiento en Z
P(:,:,3) = [0; 0; l3];
% La matriz de rotación es la identidad (0°)
R(:,:,3) = eye(3);

%---------------------------------------------------------------------------
% Construcción de las matrices de transformación homogénea
%---------------------------------------------------------------------------
Vector_Zeros = zeros(1, 3);  % Fila [0 0 0] para completar la matriz 4x4

% Inicializamos las matrices locales y globales
A(:,:,GDL)  = simplify([R(:,:,GDL) P(:,:,GDL); Vector_Zeros 1]);
T(:,:,GDL)  = simplify([R(:,:,GDL) P(:,:,GDL); Vector_Zeros 1]);
PO(:,:,GDL) = P(:,:,GDL);    % Posición del efector final
RO(:,:,GDL) = R(:,:,GDL);    % Orientación final

% Se calculan las transformaciones globales para cada eslabón
for i = 1:GDL
    A(:,:,i) = simplify([R(:,:,i) P(:,:,i); Vector_Zeros 1]);
    try
        T(:,:,i) = T(:,:,i-1)*A(:,:,i);
    catch
        T(:,:,i) = A(:,:,i);
    end
    T(:,:,i) = simplify(T(:,:,i));
    RO(:,:,i) = T(1:3,1:3,i);
    PO(:,:,i) = T(1:3,4,i);
end

%---------------------------------------------------------------------------
% Cálculo de Jacobianos y Velocidades
%---------------------------------------------------------------------------
% Para juntas prismáticas, la velocidad angular es cero.
% Se calcula el Jacobiano lineal para cada eslabón.

% --- Eslabón 3 ---
Jv_a3(:,GDL) = PO(:,:,GDL);  % Inicializa Jacobiano lineal
Jw_a3(:,GDL) = PO(:,:,GDL);  % Inicializa Jacobiano angular (no relevante en
prismáticas)

for k = 1:GDL
    if RP(k)==0
        % Caso de junta rotacional (no se usará aquí)
        try
```

```
            Jv_a3(:,k) = cross(RO(:,3,k-1), PO(:,:,GDL)-PO(:,:,k-1));
            Jw_a3(:,k) = RO(:,3,k-1);
        catch
            Jv_a3(:,k) = cross([0;0;1], PO(:,:,GDL));
            Jw_a3(:,k) = [0;0;1];
        end
    else
        % Para juntas prismáticas se utiliza el eje de traslación
(generalmente z)
        try
            Jv_a3(:,k) = RO(:,3,k-1);
        catch
            Jv_a3(:,k) = [0;0;1];
        end
        Jw_a3(:,k) = [0;0;0];
    end
end
Jv_a3 = simplify(Jv_a3);
Jw_a3 = simplify(Jw_a3);
Jac3 = [Jv_a3; Jw_a3];
Jacobiano3 = simplify(Jac3);

disp('Velocidad lineal obtenida mediante el Jacobiano lineal del Eslabón 3');
V3 = simplify(Jv_a3 * Qp);
pretty(V3);
disp('Velocidad angular obtenida mediante el Jacobiano angular del Eslabón
3');
W3 = simplify(Jw_a3 * Qp);
pretty(W3);

% --- Eslabón 2 (considera las dos primeras juntas) ---
Jv_a2(:,GDL-1) = PO(:,:,GDL-1);
Jw_a2(:,GDL-1) = PO(:,:,GDL-1);
for k = 1:GDL-1
    if RP(k)==0
        try
            Jv_a2(:,k) = cross(RO(:,3,k-1), PO(:,:,GDL-1)-PO(:,:,k-1));
            Jw_a2(:,k) = RO(:,3,k-1);
        catch
            Jv_a2(:,k) = cross([0;0;1], PO(:,:,GDL-1));
            Jw_a2(:,k) = [0;0;1];
        end
    else
        try
            Jv_a2(:,k) = RO(:,3,k-1);
        catch
            Jv_a2(:,k) = [0;0;1];
        end
        Jw_a2(:,k) = [0;0;0];
    end
```

```
end
Jv_a2 = simplify(Jv_a2);
Jw_a2 = simplify(Jw_a2);
Jac2 = [Jv_a2; Jw_a2];
Jacobiano2 = simplify(Jac2);

disp('Velocidad lineal obtenida mediante el Jacobiano lineal del Eslabón 2');
V2 = simplify(Jv_a2 * Qp(1:2));
pretty(V2);
disp('Velocidad angular obtenida mediante el Jacobiano angular del Eslabón
2');
W2 = simplify(Jw_a2 * Qp(1:2));
pretty(W2);

% --- Eslabón 1 (solo la primera junta) ---
Jv_a1(:,GDL-2) = PO(:,:,GDL-2);
Jw_a1(:,GDL-2) = PO(:,:,GDL-2);
for k = 1:GDL-2
    if RP(k)==0
        try
            Jv_a1(:,k) = cross(RO(:,3,k-1), PO(:,:,GDL-2)-PO(:,:,k-1));
            Jw_a1(:,k) = RO(:,3,k-1);
        catch
            Jv_a1(:,k) = cross([0;0;1], PO(:,:,GDL-2));
            Jw_a1(:,k) = [0;0;1];
        end
    else
        try
            Jv_a1(:,k) = RO(:,3,k-1);
        catch
            Jv_a1(:,k) = [0;0;1];
        end
        Jw_a1(:,k) = [0;0;0];
    end
end
Jv_a1 = simplify(Jv_a1);
Jw_a1 = simplify(Jw_a1);
Jac1 = [Jv_a1; Jw_a1];
Jacobiano1 = simplify(Jac1);

disp('Velocidad lineal obtenida mediante el Jacobiano lineal del Eslabón 1');
V1 = simplify(Jv_a1 * Qp(1:1));
pretty(V1);
disp('Velocidad angular obtenida mediante el Jacobiano angular del Eslabón
1');
W1 = simplify(Jw_a1 * Qp(1:1));
pretty(W1);

%----------------------------------------------------------------------
% Cálculo de la Energía Cinética
```

```
%-----------------------------------------------------------------------
% Se sustituyen las longitudes por las distancias al centro de masa
P01 = subs(P(:,:,1), l1, lc1);
P12 = subs(P(:,:,2), l2, lc2);
P23 = subs(P(:,:,3), l3, lc3);

% Matrices de inercia para cada eslabón
I1 = [Ixx1 0 0; 0 Iyy1 0; 0 0 Izz1];
I2 = [Ixx2 0 0; 0 Iyy2 0; 0 0 Izz2];
I3 = [Ixx3 0 0; 0 Iyy3 0; 0 0 Izz3];

% Cálculo de la energía cinética de cada eslabón
% La velocidad total en cada eslabón incluye el efecto del desplazamiento
del centro de masa

% Eslabón 1
V1_Total = V1 + cross(W1, P01);
K1 = (1/2 * m1 * (V1_Total)') * (V1_Total) + (1/2 * W1') * (I1 * W1);
disp('Energía Cinética en el Eslabón 1');
K1 = simplify(K1);
pretty(K1);

% Eslabón 2
V2_Total = V2 + cross(W2, P12);
K2 = (1/2 * m2 * (V2_Total)') * (V2_Total) + (1/2 * W2') * (I2 * W2);
disp('Energía Cinética en el Eslabón 2');
K2 = simplify(K2);
pretty(K2);

% Eslabón 3
V3_Total = V3 + cross(W3, P23);
K3 = (1/2 * m3 * (V3_Total)') * (V3_Total) + (1/2 * W3') * (I3 * W3);
disp('Energía Cinética en el Eslabón 3');
K3 = simplify(K3);
pretty(K3);

% Energía cinética total del robot
K_Total = simplify(K1 + K2 + K3);
disp('Energía Cinética Total');
pretty(K_Total);

%-----------------------------------------------------------------------
% Cálculo de la Energía Potencial
%-----------------------------------------------------------------------
% Se obtienen las alturas relevantes (se pueden escoger según el sistema de
referencia)
h1 = P01(1);  % Por ejemplo, altura respecto al eje x
h2 = P12(3);  % Altura respecto al eje z
h3 = P23(2);  % Altura respecto al eje y
```

```
U1 = m1 * g * h1;
U2 = m2 * g * h2;
U3 = m3 * g * h3;

% Energía potencial total
U_Total = U1 + U2 + U3;

%------------------------------------------------------------------------
% Cálculo del Lagrangiano y Modelo de Energía
%------------------------------------------------------------------------
Lagrangiano = simplify(K_Total - U_Total);
disp('Lagrangiano L = K - U');
pretty(Lagrangiano);

H = simplify(K_Total + U_Total);
disp('Función de Energía Total H = K + U');
pretty(H);

toc;  % Finaliza cronometraje
```

```
Velocidad lineal obtenida mediante el Jacobiano lineal del Eslabón 3
/ - th2p(t) (l3 sin(#1) + l2 sin(#2)) - th1p(t) (l1 sin(th1(t)) + l3 sin(#1) + l2 sin(#2)) - l3 sin(#1) th
|
|  th2p(t) (l3 cos(#1) + l2 cos(#2)) + th1p(t) (l1 cos(th1(t)) + l3 cos(#1) + l2 cos(#2)) + l3 cos(#1) th3
|
\                                                       0

where

   #1 == th1(t) + th2(t) + th3(t)

   #2 == th1(t) + th2(t)
Velocidad angular obtenida mediante el Jacobiano angular del Eslabón 3
/              0              \
|                             |
|              0              |
|                             |
\ th1p(t) + th2p(t) + th3p(t) /
Velocidad lineal obtenida mediante el Jacobiano lineal del Eslabón 2
/ - th1p(t) (l1 sin(th1(t)) + l2 #1) - l2 #1 th2p(t) \
|                                                    |
|  th1p(t) (l1 cos(th1(t)) + l2 #2) + l2 #2 th2p(t)  |
|                                                    |
\                         0                          /

where

   #1 == sin(th1(t) + th2(t))

   #2 == cos(th1(t) + th2(t))
Velocidad angular obtenida mediante el Jacobiano angular del Eslabón 2
/         0         \
|                   |
|         0         |
|                   |
\ th1p(t) + th2p(t) /
Velocidad lineal obtenida mediante el Jacobiano lineal del Eslabón 1
```

```
/ -l1 sin(th1(t)) th1p(t) \
|                          |
|  l1 cos(th1(t)) th1p(t) |
|                          |
\            0            /
Velocidad angular obtenida mediante el Jacobiano angular del Eslabón 1
/    0    \
|         |
|    0    |
|         |
\ th1p(t) /
Energía Cinética en el Eslabón 1
              2            2     ______           __                           2           2
Izz1 |th1p(t)|    |th1p(t)|  cos(th1(t) - th1(t)) m1 (l1 + lc1) (lc1 |l1|  + l1 |lc1| )
--------------- + --------------------------------------------------------------------
       2                                         2 l1 lc1
Energía Cinética en el Eslabón 2
__                                                                                     _______          __
m2 (th1p(t) (l1 sin(th1(t)) + l2 sin(#4)) + l2 sin(#4) th2p(t) + lc2 sin(th2(t)) #1) (th1p(t) (sin(#3) l2
-----------------------------------------------------------------------------------------------------------
                                                                                   2

where

   #1 == th1p(t) + th2p(t)

         _______   _______
   #2 == th1p(t) + th2p(t)

         ______   ______
   #3 == th1(t) + th2(t)

   #4 == th1(t) + th2(t)
Energía Cinética en el Eslabón 3
        / _______   _______   _______ \   __  _______                _______  __    _______              ___
        | th1p(t)   th2p(t)   th3p(t) |   m3 (th1p(t) (#5 + #8 + sin(th1(t)) l1) + th2p(t) (#5 + #8) + th3
Izz2 #1 | ------- + ------- + ------- | + -----------------------------------------------------------------
        \    2         2         2    /

where

   #1 == th1p(t) + th2p(t) + th3p(t)

         _______   _______   _______
   #2 == th1p(t) + th2p(t) + th3p(t)

   #3 == th1(t) + th2(t) + th3(t)

   #4 == th1(t) + th2(t)

                 __
   #5 == sin(#9) l2

                 __
   #6 == cos(#9) l2

                  __
   #7 == cos(#10) l3

                  __
   #8 == sin(#10) l3

         ______   ______
   #9 == th1(t) + th2(t)
```

```
          ______   ______   ______
   #10 == th1(t) + th2(t) + th3(t)
Energía Cinética Total
                   /             _______ \   __
Izz1 #11           |             th3p(t) |   m2 (th1p(t) (#8 + l2 sin(#10)) + l2 sin(#10) th2p(t) + lc2 si
-------- + Izz2 #1 | #13 + #12 + ------- | + ------------------------------------------------------------
    2              \                2    /

where

   #1 == th1p(t) + th2p(t) + th3p(t)

         _______   _______   _______
   #2 == th1p(t) + th2p(t) + th3p(t)

   #3 == th1p(t) + th2p(t)

   #4 == th1(t) + th2(t) + th3(t)

         _______   _______
   #5 == th1p(t) + th2p(t)

             ______  __
   #6 == sin(th1(t)) l1

             ______  __
   #7 == cos(th1(t)) l1

   #8 == l1 sin(th1(t))

   #9 == l1 cos(th1(t))

   #10 == th1(t) + th2(t)

                   2
   #11 == |th1p(t)|

          _______
          th2p(t)
   #12 == -------
             2

          _______
          th1p(t)
   #13 == -------
             2

                   __
   #14 == cos(#18) l2

                   __
   #15 == sin(#18) l2

                   __
   #16 == cos(#19) l3

                   __
   #17 == sin(#19) l3

          ______   ______
   #18 == th1(t) + th2(t)
```

$\#19 == \overline{\text{th1(t)}} + \overline{\text{th2(t)}} + \overline{\text{th3(t)}}$

U1 = $g\,\mathrm{lc}_1\,m_1 \sin(\mathrm{th}_1(t))$

U2 = $g\,\mathrm{lc}_2\,m_2 \sin(\mathrm{th}_2(t))$

U3 = $g\,\mathrm{lc}_3\,m_3 \sin(\mathrm{th}_3(t))$

U_Total = $g\,\mathrm{lc}_1\,m_1 \sin(\mathrm{th}_1(t)) + g\,\mathrm{lc}_2\,m_2 \sin(\mathrm{th}_2(t)) + g\,\mathrm{lc}_3\,m_3 \sin(\mathrm{th}_3(t))$

Lagrangiano(t) =

$$\frac{\mathrm{Izz}_1\,\sigma_{11}}{2} + \mathrm{Izz}_2\,\sigma_1\,\left(\sigma_{13} + \sigma_{12} + \frac{\overline{\mathrm{th3p}(t)}}{2}\right) + \frac{\overline{m_2}\,\left(\mathrm{th1p}(t)\,\left(\sigma_8 + l_2\sin(\sigma_{10})\right) + l_2\sin(\sigma_{10})\,\mathrm{th2p}(t) + \mathrm{lc}_2\sin(\mathrm{th}_2(t)\right.}{2}$$

where

$$\sigma_1 = \mathrm{th1p}(t) + \mathrm{th2p}(t) + \mathrm{th3p}(t)$$

$$\sigma_2 = \overline{\mathrm{th1p}(t)} + \overline{\mathrm{th2p}(t)} + \overline{\mathrm{th3p}(t)}$$

$$\sigma_3 = \mathrm{th1p}(t) + \mathrm{th2p}(t)$$

$$\sigma_4 = \mathrm{th}_1(t) + \mathrm{th}_2(t) + \mathrm{th}_3(t)$$

$$\sigma_5 = \overline{\mathrm{th1p}(t)} + \overline{\mathrm{th2p}(t)}$$

$$\sigma_6 = \sin\left(\overline{\mathrm{th}_1(t)}\right)\,\overline{l_1}$$

$$\sigma_7 = \cos\left(\overline{\mathrm{th}_1(t)}\right)\,\overline{l_1}$$

$$\sigma_8 = l_1\sin(\mathrm{th}_1(t))$$

$$\sigma_9 = l_1\cos(\mathrm{th}_1(t))$$

$$\sigma_{10} = \mathrm{th}_1(t) + \mathrm{th}_2(t)$$

$$\sigma_{11} = |\mathrm{th1p}(t)|^2$$

$$\sigma_{12} = \frac{\overline{\mathrm{th2p}(t)}}{2}$$

$$\sigma_{13} = \frac{\overline{\mathrm{th1p}(t)}}{2}$$

$$\sigma_{14} = \cos(\sigma_{18})\,\overline{l_2}$$

$$\sigma_{15} = \sin(\sigma_{18})\,\overline{l_2}$$

$$\sigma_{16} = \cos(\sigma_{19})\,\overline{l_3}$$

$$\sigma_{17} = \sin(\sigma_{19})\,\overline{l_3}$$

$$\sigma_{18} = \overline{\mathrm{th}_1(t)} + \overline{\mathrm{th}_2(t)}$$

H(t) =

$$\frac{\mathrm{Izz}_1\,\sigma_{11}}{2} + \mathrm{Izz}_2\,\sigma_1\,\left(\sigma_{13} + \sigma_{12} + \frac{\overline{\mathrm{th3p}(t)}}{2}\right) + \frac{\overline{m_2}\,\left(\mathrm{th1p}(t)\,\left(\sigma_8 + l_2\sin(\sigma_{10})\right) + l_2\sin(\sigma_{10})\,\mathrm{th2p}(t) + \mathrm{lc}_2\sin(\mathrm{th}_2(t)) \right.}{2}$$

where

$$\sigma_1 = \mathrm{th1p}(t) + \mathrm{th2p}(t) + \mathrm{th3p}(t)$$

$$\sigma_2 = \overline{\mathrm{th1p}(t)} + \overline{\mathrm{th2p}(t)} + \overline{\mathrm{th3p}(t)}$$

$$\sigma_3 = \mathrm{th1p}(t) + \mathrm{th2p}(t)$$

$$\sigma_4 = \mathrm{th}_1(t) + \mathrm{th}_2(t) + \mathrm{th}_3(t)$$

$$\sigma_5 = \overline{\mathrm{th1p}(t)} + \overline{\mathrm{th2p}(t)}$$

$$\sigma_6 = \sin\left(\overline{\mathrm{th}_1(t)}\right)\,\overline{l_1}$$

$$\sigma_7 = \cos\left(\overline{\mathrm{th}_1(t)}\right)\,\overline{l_1}$$

$$\sigma_8 = l_1\sin(\mathrm{th}_1(t))$$

$$\sigma_9 = l_1\cos(\mathrm{th}_1(t))$$

$$\sigma_{10} = \mathrm{th}_1(t) + \mathrm{th}_2(t)$$

$$\sigma_{11} = |\mathrm{th1p}(t)|^2$$

$$\sigma_{12} = \frac{\overline{\mathrm{th2p}(t)}}{2}$$

$$\sigma_{13} = \frac{\overline{\mathrm{th1p}(t)}}{2}$$

$$\sigma_{14} = \cos(\sigma_{18})\,\overline{l_2}$$

$$\sigma_{15} = \sin(\sigma_{18})\,\overline{l_2}$$

$$\sigma_{16} = \cos(\sigma_{19})\,\overline{l_3}$$

$$\sigma_{17} = \sin(\sigma_{19})\,\overline{l_3}$$

$$\sigma_{18} = \overline{\mathrm{th}_1(t)} + \overline{\mathrm{th}_2(t)}$$

Obtención de la Energía Cinética Total en un Robot Cartesiano de 3 GDL

Introducción

Este reporte presenta el procedimiento seguido para obtener la energía cinética total de un robot cartesiano de 3 grados de libertad (3 GDL) en el que todas las juntas son prismáticas. Dado que en este tipo de robot el movimiento es lineal, la cinemática se define mediante desplazamientos directos, y la contribución angular es nula. La metodología combina la formulación simbólica en MATLAB para obtener las expresiones generales de la cinemática y la dinámica.

Objetivos

- Definir la cinemática del robot cartesiano: Determinar las posiciones y orientaciones de cada eslabón en función de los desplazamientos lineales.
- Calcular los Jacobianos y las velocidades: Establecer la relación entre las velocidades articulares y la velocidad del efector final.
- Obtener la energía cinética total: Incluir tanto la contribución traslacional (debida al movimiento del centro de masa) como, en su caso, la rotacional.
- Determinar el Lagrangiano y la energía total: Estos modelos son esenciales para posteriores análisis dinámicos y diseño de controladores.

Metodología

Procedimiento1. Preparación del Entorno y Declaración de Variables

- Descripción:Se limpia el entorno (borrado de variables, cierre de figuras y limpieza de la consola) y se declaran las variables simbólicas correspondientes a los desplazamientos, velocidades y aceleraciones de cada junta, así como los parámetros físicos (masa, inercia, longitudes y distancias al centro de masa).

2. Definición de la Cinemática Directa

- Descripción:Se establecen las posiciones y orientaciones de cada eslabón. Para un robot cartesiano, cada articulación se asocia a un desplazamiento lineal en el eje, y se incorporan matrices de rotación específicas que reorientan el sistema según el diseño.
- Pasos:
- Articulación 1: Definición de la posición P(:,:,1)=[0;0;l1] y una rotación (-90° en el eje x) para definir la orientación.
- Articulación 2 y 3: Definición similar usando desplazamientos en Z y las rotaciones correspondientes.

3. Construcción de las Matrices de Transformación Homogénea

- Descripción:Se generan las matrices que combinan la información de posición y orientación para transformar las coordenadas desde el marco base al efector final.
- Pasos:

- Se construyen las matrices locales A(:,:,i) para cada eslabón.
- Se calculan las matrices globales T(:,:,i)mediante multiplicación secuencial.
- Se extraen las posiciones PO y orientaciones RO.

4. Cálculo del Jacobiano y las Velocidades

- Descripción:Para cada eslabón se calcula el Jacobiano lineal que relaciona los desplazamientos prismáticos con la velocidad del efector. En este caso, la velocidad angular es nula (juntas prismáticas).
- Pasos:
- Se calcula el Jacobiano para el eslabón 3 usando todas las juntas; para el eslabón 2 se usan las dos primeras y para el eslabón 1 sólo la primera.
- Se multiplica cada Jacobiano por el vector de velocidades Qp para obtener las velocidades lineales V1,V2,V3

5. Cálculo de la Energía Cinética

- Descripción:Se obtiene la energía cinética de cada eslabón considerando la contribución traslacional (movimiento del centro de masa) y la rotacional. Para determinar la posición del centro de masa se sustituyen las longitudes máximas por las distancias al centro de masa.
- Pasos:
- Se sustituye li por lci en la posición para cada eslabón.
- Se calcula la velocidad total VTotal=V+cross(W,Pcm)
- Se determina la energía cinética de cada eslabón mediante: K= 0.5m * ∥VTotal∥^2 + 0.5 W*I*W
- Se suman las energías cinéticas parciales para obtener KTotal

6. Cálculo de la Energía Potencial y del Lagrangiano

- **Descripción:**La energía potencial se calcula a partir de la altura del centro de masa respecto a un eje (según el sistema de referencia). Luego se define el Lagrangiano como la diferencia entre la energía cinética y la potencial, y la energía total como la suma de ambas.
- **Pasos:**
- Se determina la altura h para cada eslabón.
- Se calcula U=mgh para cada uno y se suma para obtener UTotal
- Se establece: L=KTotal − UTotal y H=KTotal + UTotal

Conclusión

El análisis presentado integra la descripción de la cinemática y la dinámica de un robot cartesiano de 3 GDL, en el que se consideran juntas prismáticas. Cada paso desde la definición de las variables y la construcción de las matrices de transformación hasta el cálculo de los Jacobianos, las velocidades y finalmente las energías cinética y potencial es fundamental para comprender el comportamiento dinámico del sistema. Este procedimiento es la base para el diseño de controladores y la simulación del movimiento en sistemas robóticos cartesianos.